TreeNode4.java

```java
public class TreeNode4 extends TreeNode {

  /**
   * Construct this with the given id and child TreeNode references.
   */
  public TreeNode4(int id, TreeNode child1, TreeNode child2,
                   TreeNode child3, TreeNode child4) {
    super(id);
    this.child1 = child1;
    this.child2 = child2;
    this.child3 = child3;
    this.child4 = child4;
  }

  /**
   * Return the String representation of this subtree, which is the String
   * value of its ID, followed on the next three indented lines by the
   * recursive toString of its three children. See the documentation for
   * <a href="TreeNode.html#toString()">TreeNode.toString()</a> for a general
   * description the way trees are represented as strings.
   */
  public String toString(int level) {
    String indent = "";
    for (int i = 0; i < level; i++) {
      indent += " ";
    }
    return symPrint(id) + "\n" +
           indent + "* + (child1 == null ? "null" : child1.toString(level+i)) + "\n" +
           indent + "* + (child2 == null ? "null" : child2.toString(level+i)) + "\n" +
           indent + "* + (child3 == null ? "null" : child3.toString(level+i)) + "\n" +
           indent + "* + (child4 == null ? "null" : child4.toString(level+i));

  }

  /**
   * Reference to the left child of this node. */
  public TreeNode child1;

  /**
   * Reference to the first middle (or second, or third-from-the-last) child
   * of this node. */
  public TreeNode child2;

  /**
   * Reference to the second middle (or third, or next-to-the-last) child of
   * this node. */
  public TreeNode child3;

  /**
   * Reference to the right (or fourth, or last, or rightmost, or
   * whatever-the-heck-you-want-to-call-it) child of this node. */
  public TreeNode child4;
}
```

****

* TreeNode4 extends TreeNode by adding four child components, which are
  * references other TreeNodes. Hence, TreeNode4 is used to represent
  * quartinary syntactic constructs in a parse tree.
* */

public class TreeNode4 extends TreeNode {

  /**
   * Construct this with the given id and child TreeNode references.
   */
  public TreeNode4(int id, TreeNode child1, TreeNode child2,
                   TreeNode child3, TreeNode child4) {
    super(id);
    this.child1 = child1;
    this.child2 = child2;
    this.child3 = child3;
    this.child4 = child4;
  }

  /**
   * Return the String representation of this subtree, which is the String
   * value of its ID, followed on the next three indented lines by the
   * recursive toString of its three children. See the documentation for
   * <a href="TreeNode.html#toString()">TreeNode.toString()</a> for a general
   * description the way trees are represented as strings.
   */
  public String toString(int level) {
    String indent = "";
    for (int i = 0; i < level; i++) {
      indent += " ";
    }
    return symPrint(id) + "\n" +
           indent + "* + (child1 == null ? "null" : child1.toString(level+i)) + "\n" +
           indent + "* + (child2 == null ? "null" : child2.toString(level+i)) + "\n" +
           indent + "* + (child3 == null ? "null" : child3.toString(level+i)) + "\n" +
           indent + "* + (child4 == null ? "null" : child4.toString(level+i));

  }

  /**
   * Reference to the left child of this node. */
  public TreeNode child1;

  /**
   * Reference to the first middle (or second, or third-from-the-last) child
   * of this node. */
  public TreeNode child2;

  /**
   * Reference to the second middle (or third, or next-to-the-last) child of
   * this node. */
  public TreeNode child3;

  /**
   * Reference to the right (or fourth, or last, or rightmost, or
   * whatever-the-heck-you-want-to-call-it) child of this node. */
  public TreeNode child4;
```